HOLON: A Web-Based Framework for Fostering Guideline Applications

by

Barry G. Silverman[1,6], PhD, Khalid Moidu,[2], MD, PhD, Bernadette E. Clemente[3], MS, Larry Reis[4], MS,
David Ravichandar[5], MS, Charles Safran[6], MD, MS
1-George Washington University, 2-Norwalk Hospital, 3-Concept Five Technologies, 4-Wizdom Systems, 5-Oracle Corp., 6- Center for Clinical Computing, Harvard Medical School
Contact: barry@seas.gwu.edu
March 1997

HOLON is a research and development effort in extending middleware in the healthcare field to support application development, in general, and guideline applications, in particular. This framework makes use of open standards for architecture, software, guideline KBs, clinical repository models, information encodings, and intelligent system modules and agents. By pursuing the use of such standards in our middleware components, we hope eventually to maximize reusability of the HOLON framework by others who also adhere to these open standards. This research reflects lessons learned about the extensions needed in these standards if healthcare middleware frameworks are to transparently support application developers and their users over the web.

1) INTRODUCTION

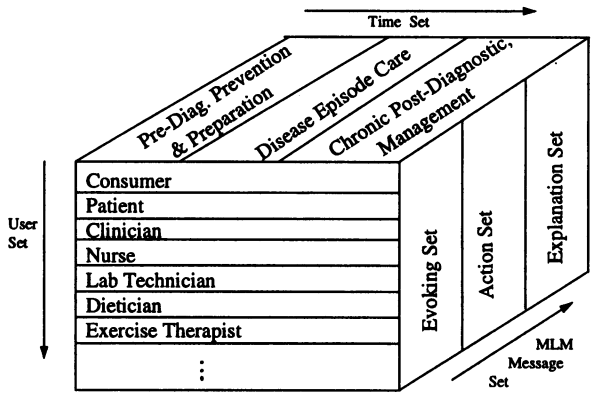
The Health Object Library ONline (HOLON) is an object-oriented, healthcare library and middleware utility that users and applications can plug into to collaboratively and securely gain access to globally distributed, heterogeneous information. Users, here, is a broad term that covers patients, consumers, health care providers, administrators, analysts, and the like. HOLON is being developed jointly by 14 organizations, several of which are the co-authors'.

In the near future, virtual health care organizations will utilize telecommunication supported networks to connect distributed communities of professionals and consumers. Already, non-portable, heterogeneous patient data and isolated applications are becoming less tolerated in view of progress in health informatics standards, the information highway, and a proliferating set of access devices. Also, due to cost and quality control concerns, inspection, rework, and paper intensive processes will give way to reusable electronic records and information sets that are machine-checked for omissions and to get the decisions right the first time. To meet such expectations, applications developers are attempting to shift formerly closed, isolated applications toward open, rapidly customizable software.

The purpose of HOLON is to research and develop components that will help such application developers and their user communities. To focus this research, HOLON initially is limiting itself to one domain (diabetes mellitus). Also, heavy emphasis is being placed on context and situationally triggered decision support applications, such as for the guidelines area.

A guideline is essentially a set of knowledge base directives for a process, or the next step in the process of care. A compilation has been made of all available guidelines for the care of a diabetic and an ontology and classification of these guidelines is currently underway. As seen in Figure 1, at the top level, the ontology offers a view of the users, the time in the clinical stage of the disease, and the types of message sets that application developers wish to move between users. In the Arden Syntax Standard from ASTM, each guideline is an independent medical logic module (MLM). These molecular MLM applications are evoked by events such as updates to medical records. Once evoked, an MLM recommends actions such as alerts, ToDo items, reminders, and the like. It also may precipitate educational and explanation applications and materials relevant to any of the members of the user set. The guideline arena thus serves as a rich area on which to focus initial HOLON requirements specifications and development effort.

Figure 1 - Dimensions of Support Required for Guideline Applications: Problem Set - Diabetes Mellitus, Systemic Complications of Weight and Blood Sugar

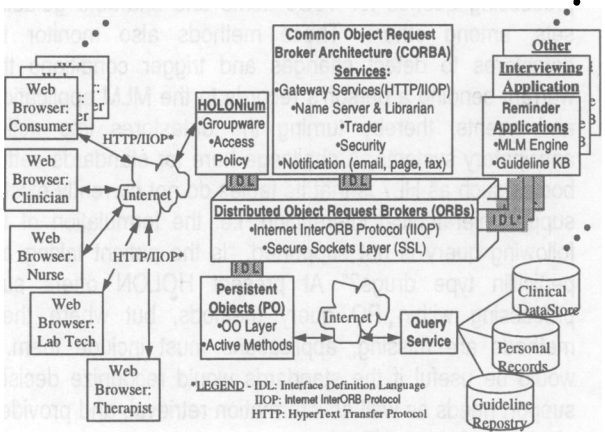


2) THE HOLON FRAMEWORK

The HOLON architecture uses principles of modern software engineering. In Figure 2, one can see multiple clients on the left, data servers on the right, and middleware connecting the two. Applications, legacy patient record systems (once wrapped) and other components can be readily plugged into this architecture just by registering them

with the common object services. Returning to the details of Figure 2, there are six major components to explain:

Figure 2 - The HOLON Framework and its Open Standards Components That Connect Guideline Applications to User Clients over the Web and to Health Records Via Object Oriented Technology and Services



Clients -- The initial client design target is a web browser and email reader on a desktop PC, although our sponsor also eventually expects interactive TV and beeper service. The client includes (see Figure 2) a web browser, web-enabled email viewer, and a terminal stay-resident (TSR) alerter component that parses email and alerts users when high priority health items have arrived. The web browser is the main route through which applets (1) interview users for personal health information and patient record data, (2) elicit "user preferences" for health plans, recurring ToDo items, and reminder schemes, and (3) send action (ToDos) and explanation and education message sets. All three elements of the client make use of the Java virtual machine for executing applet bytecodes, and thereby maximize platform independence, multi-threading, and security of local resources. Access is further constrained by password, PIN, and enterprise privacy control policy settings.

Groupware -- HOLON's users are treated as a community. To support this, HOLON incorporates an email router with a forward and CC list for user-designated members to share MLM triggered ToDo and Reminder items. Also, to foster inter-user dialog, HOLON offers a state of the art groupware environment that recognizes CORBA enabled Java applets. Called HOLONium, this environment offers virtual meeting rooms, a whiteboard for jointly viewing/marking MLM message sets, chat space, and CUSeeme video-conferencing facilities. HOLONium also enforces end user access privileges, authentication, credential request, and security audit. We are currently using HOLONium in our testbed at the Beth Israel-Deaconess Medical Center in Boston, MA and in Norwalk Hospital, CT.

Distributed ORBs -- HOLON uses the Common Object Request Broker Architecture (CORBA) as the standard for distributed object management [4]. CORBA provides the basic messaging mechanisms needed by objects to communicate with one another across heterogeneous languages, platforms, and networks. An Object Request Broker (ORB) enables a client object to transparently invoke a method on a server object by locating the server object that can implement the method, passing it the parameters, and returning the results. Each server object is described using Interface Definition Language (IDL), a language-neutral, declarative way to specify an object's boundaries and its interfaces with potential clients. Rather than connecting directly to server objects, HOLON clients interact with servers via Visigenic's VisiBroker using the HyperText Transfer Protocol (HTTP) and Internet Inter-ORB Protocol (IIOP).

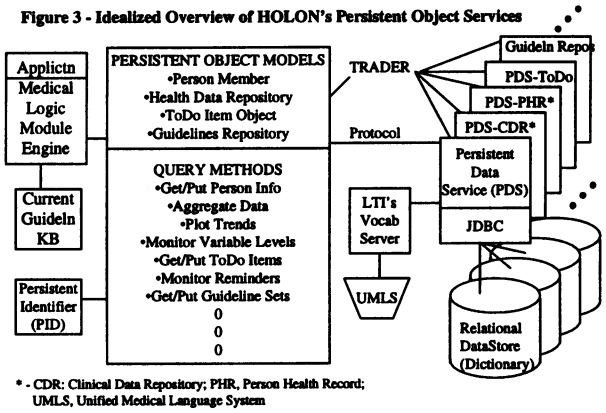
CORBA Services and Facilities -- The CORBA specification includes a set of services common to object systems such as naming, trading, secure communications, and notification. Unfortunately, few of these services are fully available off the shelf at this time. Our partner, Concept Five is developing several of these services as required by other HOLON components.

As one example, while the CORBA security service specification exists, many supporting security products are still under development. Since most application developers will be consumed with their application-specific focus, we are insulating their application code interfaces from these evolving security features. HOLONium does this for access. For communication security we also have embedded a Secure Socket Layer (SSL) within HOLON's inter-component communication protocol. The security products may mature, evolve, and become fully compliant without impact on application code.

In addition to being unfinished, the various CORBA service specifications are generic. To tailor them to the middleware needs of an industry like healthcare requires the addition of vertical domain facilities. An example of this is the inability of the name service alone to support patient identification and record locating needs. CORBAmed, a healthcare-specific interest group is currently calling for requests for proposals for a "master patient index (MPI)." An MPI is intended to address the protocol for identifying patients across physical and organizational boundaries. The naming service is inadequate fo his task since it only binds a unique name to a single object. As a subset of the MPI topic, in order to locate a given patient's records distributed across physical and organizational boundaries, HOLON has been developing a "health record librarian." This librarian maintains a set of pointers to the set of health records registered for each patient.. It also can maintain pointers to other distributed repository items, such as clinician ToDo items, diverse organizations' guideline sets, and so on. The Librarian uses the Lightweight Directory Access Protocol (LDAP) for X.500 Directory Services to store these pointers. This Librarian "facility" is essential if HOLON is to support the distributed user set mentioned earlier. It also centralizes control as required by security concerns

Persistent Object Service(POS) -- This common object service is a principal integrating component between applications and the distributed, heterogeneous data such as clinical data repositories (CDR), personal health records (PHR), ToDo item and reminder information generated by the consumer or the MLM guideline engine. HOLON provides this integration via the components shown in Figure 3 and discussed below. The CORBA specification for POS includes interfaces for the Persistent Object (PO), the Persistent Identifier (PID) string that specifies one or more locations of the persistent data store for that PO, the Persistent Data Service (PDS) as the object responsible for moving data in/out of the datastore, and the PO Manager or Trader for routing calls from the PO to the right PDS using the PID.

Figure 3 - Idealized Overview of HOLON's Persistent Object Services



* - CDR: Clinical Data Repository; PHR, Person Health Record; UMLS, Unified Medical Language System

In practice, HOLON found it necessary to extend this POS specification in numerous ways that are also depicted in Figure 3. First, to effectively insulate applications, the PO layer requires a domain object model and set of query methods. To support this, HOLON is deploying a set of Persistent Object Models (POMs) that delineate the structure of items used by guideline and other applications such as the person-member object, the Clinical Data Repository (CDR), the Personal Health Record (PHR), the ToDo and reminder item objects, and distributed guideline repositories, among others. Each of these models attempts to conform to health informatics or other relevant standards that foster application recognition of class and attribute structure and terminology. An obstacle is that many of the standards groups have not finished the object models HOLON needs at this layer. For example, the CDR model includes the ANSI HL7 reference model, DICOM, and various IEEE standards. Various members of the HOLON team are aggressively extending these standards where repository items are unspecified, such as in the areas of PHR, ToDo Items, and guideline repositories. This means as new standards are generated, HOLON's PO models will need to be updated, yet past versions must be maintained until interfaces to applications making use of outdated PO models can be converted. There is no way around this.

A second feature of HOLON's POS is that it incorporates domain specific PO Query Methods. These methods perform a wide range of high level I/O functions such as aggregating data, assembling trends, monitoring variables, and processing queries for ToDo items and sharable guideline sets, among others. These methods also monitor the datastores to detect changes and trigger conditions that warrant sending a person's records to the MLM applications and agents, thereby turning the datastores into active, anticipatory systems. A challenge here for standards setting bodies such as HL7 is that its tables do not currently exist to support hierarchical inferencing, i.e. the formulation of the following query is not supported, "Is the patient taking any penicillin type drugs?" At present HOLON offers such processing within PO query methods, but where these methods are missing, applications must include them. It would be useful if the standards would recognize decision support needs as well as information retrieval, and provide a more complete specification.

A third extension of the POS by HOLON concerns the query server within the PDSs. The query server standard within CORBA was adopted in the summer of 1996, but that standard adheres to the ISO's SQL3 and Object Query Language (OQL) standards that are not yet widely available in current database products. For these reasons, we found we had to implement our own query servers. We deviate from the OMG Query Server and use a Java DataBase Connectivity (JDBC) standard set of drivers for connection outside the ORB. That is why earlier Figure 2 shows the distributed databases in our current testbed connecting through the data server, rather than directly to the ORB.

A fourth and final extension we will discuss here concerns the inclusion of a vocabulary server and thesaurus capability. These should help reduce lexical translation difficulties between locally derived datastores and centrally defined PO models and methods. Lexical Technologies Inc.'s vocabulary server will be piloted for this purpose as a subcontract to BIDMC, one of the HOLON consortium members. While the vocabulary sets that need to be referenced are not under the purview of any single standard setting body, LTI supports the National Library of Medicine's Unified Medical Language System, a thesaurus that the HL7 vocabulary SIG is proposing be extended to include existing and future tables supporting the HL7 messaging standard as well as those sets of existing and future vocabularies that can enable the exchange of medical data and information in a well defined and unambiguous context.

Guideline & Reminder Server -- This server is an example of the highest level of application development support that HOLON provides. Earlier Figure 1 showed that countless guidelines applications need to be authored and deployed in the effort to improve quality management in healthcare. This can only be accomplished if industry-wide cooperation occurs where guidelines are shared and reused

after tailoring to a given site's needs. Our goal is to foster this type of sharing and reuse.

Study of guideline and reminder applications [e.g., 1, 4] quickly reveals four components are needed to foster this sharing: an Arden Syntax elicitor/editor, a guideline repository, an application builder, and a reminder engine. For the latter two items, in the Fall of 1996 (see next section) HOLON adopted a web-based, full-featured reminder engine called R2Do2 that uses a Java version of the CLIPS engine: see [7]. A more daunting challenge lies in finding or creating a suitable editor. The ASTM's Arden Syntax for Medical Logic Modules (MLMs) requires maintenance, library, and knowledge sections, each of which must follow a certain structure. This structure and editors based on it are insufficient to foster guideline or MLM sharing between developers.

Several extensions to this structuring are provided by HOLON that may be useful here. For example, the Arden Syntax Standard leaves the precise data access syntax unstandardized, with the idea that it would be written locally because of the different IS environments. Yet the local rewriting of that section is time consuming, error prone, and requires person to person consultation between someone at the author's institution and the person attempting to reuse it. The Persistent Object Services of HOLON provide an alternative where decision logic can be dynamically linked to information access components that can map local syntactic or semantic variations into "standard or commonly used forms" of representing clinical information or concepts of medical knowledge. The existence or occurrence of a coded event (e.g., 'a new DT booster was administered') can be built into the structure of MLM statements for evoking or action messages. To date, our approach seems feasible and we think it would prove a viable way to extend the Arden Standard.

Another example is the lack of standard lexicon in Arden MLMs that further precludes guideline sharing and reuse. This combines with the need for a higher level "structured English" editor that will foster guideline authoring by a wider set of users. The POS' inclusion of object templates and the vocabulary server open the door to the creation of structured-English level editors of the guideline KBs and the various rule elements (events, actions, etc.) in which users could avoid the need to code access connections or to deviate from approved repository vocabulary, attribute names, and message structures. We are currently pursuing this avenue of editor development. Once it is completed, we believe this approach, combined with the POS handling of distributed guideline repositories, will open up new, realistic opportunities for reuse and sharing.

4) CASE STUDIES and LESSONS LEARNED

To date we have undertaken two "use cases" that tested the end-to-end HOLON framework just described. The first, called the "single thread" scenario, was a case that HOLON had to cope with during the Fall of 1996. The single thread scenario involved a user and clinician on two separate web browsers interacting remotely with the HOLONium in Burlington, MA and the MLM engine running as an applet from a server in downtown Washington DC. The patient's health records were maintained by a Java object that gave persistence to this information and by implementing a JDBC/ODBC bridge to an Oracle 7 relational database in Chicago, Ill. These records were fed by Windom Health's patient interview or profiler module run from a client applet in Berkeley, CA. This was a simple test of the HOLON architecture involving only two members of the user set, about a dozen patient attributes, and only two MLMs in the guideline set. The two sample applications -- patient interview system and guideline reminder engine -- were integrated into the HOLON framework. On the front end, connection was simple since any Java or web-enabled applications can be connected to the HOLONium once it is registered. On the backend, connecting the applications required writing CGI bin server side scripts and CORBA IDL for each application, respectively. This cost those organizations about 3 man-months combined in learning curve and programming effort, plus significantly more than that from HOLON developers for both assisting the application developers and for fixing HOLON "bugs." Some of the bugs that were discovered and have since been fixed or identified as research needs include:

- an application developer's handbook has been authored
- ways to write maintenance-free CORBA Interface Definition Language (IDL) were derived (i.e., attributes omitted)
- the need for a HOLON Application Program Interface (API) shielding application developers from CORBA
- more robust HOLON service needs were identified (i.e., the Health Record Librarian and several of the persistent object services described in this paper)
- the obvious (in hindsight) need for more complete OO repository models was highlighted and raised as a priority for added HOLON funding
- the ability for application developers to independently extend the OO models and query methods in HOLON's persistent layer.

The latter finding is leading to the evaluation of how to include various CASE and OO modeling tools such as the Unified Modeling Language for OO programming, the Sedona Toolset for Oracle8, and others. As yet, application developers must work with HOLON developers to extend this layer.

The second use case involves a testbed that is ongoing as of this writing. This testbed involves four members of the user set from Figure 1, distributed at various locations around the Beth Israel Deaconess Medical Center and at a home-based site, all in Boston, MA. This time around the interview and guideline reminder engines are already HOLON components,

and the effort (assisted by the handbook) is to extend these applications by adding dozens of new interview items, about 100 new guideline rules, a few 100 new OO repository attributes, and 100s of new data fields and values. Also, a third application developer is adding a decision modeling tool as part of this testbed.

Up to this point our general framework as presented here appears robust, and we haven't identified any major new categories of challenge. In terms of known challenges, we are both excited by the synergy between the various standards, and cautious about finding ways to insulate applications from inevitable changes as the standards evolve. In terms of the synergies the HOLON framework illustrates specifics on how:

1. integrating the CORBA, HL7, and vocabulary standards, among others, can improve the prospects for reuse of MLMs created under the Arden Syntax;
2. merging the HL7 messaging standard, CDR repository models, standardized vocabulary sets, and Arden MLM structuring can improve the CORBA vertical facilities such as the Persistent Data Store standard, and the Name Server/Record Librarian;
3. Combining CORBA and Arden in a web-based environment highlights new needs for extending the HL7 messaging standard (e.g., repository models, guideline object structure, and ToDo items, among others).

5) RELATED WORK

The design of the HOLON framework has been influenced by numerous sources, only a few of which will be mentioned here. In terms of a web-enabled patient record system, the work of Kohane [3], while in PERL, pioneers many of the features that are needed at the interface. The NLM/AHCPR and Mass General Hospital [2] guideline repositories, while not MLM based and unconnected from patient records, illustrate added web display and search features of use for group discussion of guidelines in the HOLONium. Finally, the design of the HOLONium is also influenced by the collaborative world of MUSHes, MUDs, and MOOs.

A number of MLM-oriented guideline processing frameworks that connect to patient record systems exist such as Arachne [8], EON [5], and HELIOS [1], among numerous others. Many of these are richly featured in terms of guideline knowledge, MLM editors, and framework ideas. Often they are single user rather than group-focused; rich in framework prototyping but omit robust middleware services; and as yet only minimally integrate health informatics standards (e.g., the HL7 messaging standard, CDR repository models, vocabulary servers, and guideline structure standards) for overcoming the Arden-related obstacles to reuse. The EON system's developers offer the most complete set of ideas for guideline reuse, editing, and site-specific adaptation. The HELIOS environment, though now non-operational, pioneered a unification bus and security services for a hospital environment. The Arachne, and related InterMed Collaboratory, is spearheading standards for guideline authoring that, when completed, could further reduce the exchange obstacles described earlier.

Lastly, HOLON's framework is influenced by the results of standard setting bodies in the medical informatics field.

6) CONCLUDING COMMENT

The purpose of HOLON is to provide a set of reusable components that comprise a framework for fostering the rapid development of web-based applications, in general, and guideline applications, in particular. The CORBA specification partially addresses many of these items. Yet sector- and industry-specific frameworks sitting above traditional middleware are needed to complete this list and to open up the full potential for middleware to help application development. In the health care sector, a number of standards support such a framework: e.g., ANSI HL7, ASTM Arden Syntax, and DICOM, among others. Our research investigates how vertical frameworks and guidelines applications support environments can exploit some of these advances.

ACKNOWLEDGMENT

The financial support of NIST/ATP is gratefully acknowledged Inaddtion to the consortium partners identified on the front page we would like to acknowledge other HOLON consortium partners: Windom Health, Lumina, Talisman, Meta, IntelliTek, and the Koop Foundation.

REFERENCES

1.Arkad, K., et al., Integration of Data Driven Decision Support Into the HELIOS Environment, Internat'l J of Bio-Med Comp.,34 (1995):195-205

2.Barnes, M., Barnett, GO, An Architecture for a Distributed Guideline Server, SCAMC Proc., Bethesda: AMIA, 1995, pp. 233-37.

3.Kohane, I.S., et al., "Building National Electronic Medical Record Systems via the World Wide Web," JAMIA, v.3, n.3, May/Jun 1996, pp. 191-207.

4.Otte, R., et al, The Common Object Request Broker Architecture, Englewood Cliffs: Prentice Hall, 1996.

5.Musen,M., et al., "EON:A Component Based Approach to Automation of Protocol Directed Therapy," JAMIA, 3(1996):367-388.

6.Anon., Health Object Library Online (HOLON), URL=http://www.omg.org/CORBAMED/96-05-14; Gaithersburg: Koop Foundation Inc., White paper, 1996.

7.Silverman, B., et al., "Web-Based Health Care Guideline Agents: The Case of Reminders of ToDos-II (R2Do2)", Washington DC: GWU/IAI Tech Report, Jan. 1997. URL = http://iai.seas.gwu.edu/iailab/R2Do2.html

8. Deibel, S.R.A., Greenes, R.A., "An infrastructure for the development of health care information systems from distributed components," J. Amer. Soc. Information Science. 46(10): 765-771.